



下载APP



## 03 | 反爬虫的发展以及展望：我们也不是什么正经工程师

2022-01-24 DS Hunter

《反爬虫兵法演绎20讲》

课程介绍 &gt;



讲述：DS Hunter

时长 22:58 大小 21.04M



你好，我是 DS Hunter。

在上一节课里，我们讲了反爬虫方是如何对抗爬虫方这个坏人的。这节课，咱们一起看一看，在内卷之下，反爬虫真的能保证“不作恶”吗？

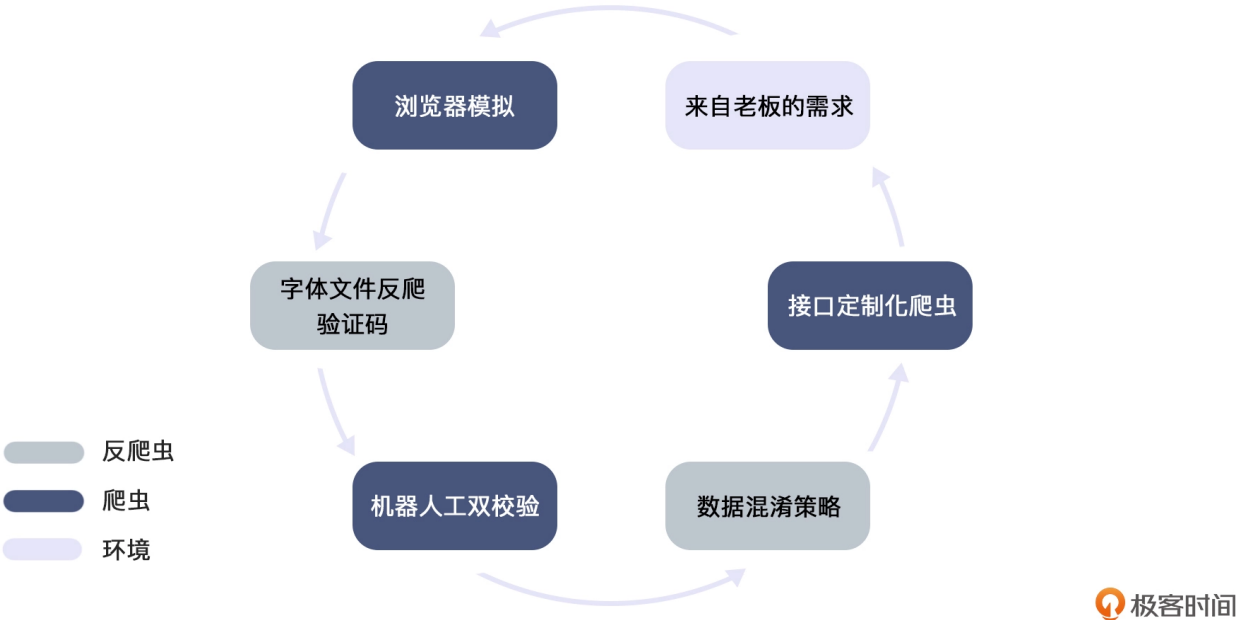
这节课是我们历史背景篇的最后一节课，我们来点轻松的，我会首先讲一讲反爬虫方和爬虫方这两方各自最通用的两个手段——验证码和浏览器模拟的历史过程，然后再用几个故事来带你看看反爬虫的近期发展。



在一些故事的结尾，我也会预言一下后续的发展。当然了，预测未来是一个不怎么靠谱的事情，失败概率极高，不过我还是来无责预测下。

## 爬虫方和反爬虫方的大招

我们这里把大招理解为以不变应万变的经典招式，它们不会固定存在于爬虫和反爬虫克制链条中的某一环节，但是，却可以无处不在。当然，我还是把它放到了链条中，因为当斗争激烈到某个程度时，大家更倾向于想起这些经典大招（下面的链条图仅供参考）。



### 低效率的浏览器模拟

我们先来看看爬虫方的进攻手段，浏览器模拟。在我们上一讲中的爬虫第一招——接口定制化部分提到的反转，指的就是这里。爬虫从全面模拟浏览器，变为只抓指定接口，现在又变回了全面模拟浏览器，转了一圈又回到了原地。因为“浏览器模拟”和验证码一样，基本上是万能的——因为被封杀的可能性最低。

是的，爬虫厌倦了无休止的检测与反检测，决定直接使用浏览器模拟请求。这个时候，内卷重新开始。但是要知道，请求价格接口只有一个请求，而浏览器请求站点，请求量会急剧飙升，拖慢网站性能。

针对浏览器爬虫，业界内并没有特别好的解决方案。以前有 canvas 指纹，现在有 DOM 指纹，核心都是**标记相同类型的浏览器**。原因是，大部分的浏览器都是批量部署的，因此环境完全一致。

canvas 指纹可以在一定程度上检测到相同类型的人，然后你可以给予封杀。但是这个并不准，甚至可以说误伤率很高。而且从描述我们就看得出，这里的封杀，并不是封杀爬虫，而是封杀“有相同特征的请求”，如果人工找到其中一个是爬虫，那么这一批就都不放过。但是我们并不在乎这种请求是否是真人。此外就是指纹也存在冒充，你要给予足够复杂的加密措施，让他拿不到原始指纹来冒充。DOM 指纹也是一样的道理，主要用于检测 Selenium 之类的平台。

如果非说有什么解决方案，那就是**瞄准性能——浏览器爬虫的死穴**，开干。浏览器爬虫性能很糟糕，接口爬虫能爬一万次的话，相同资源下浏览器爬虫能爬到一百就了不起了——实际上可能更差，这就是我说它效率低的原因。

因此，反爬方极可能会随机拖慢网站性能，来进一步影响爬虫方的效率，甚至可以随机拖慢信息加载的时间。要知道，价格并不影响 SEO（Search Engine Optimization，搜索引擎优化），就算有什么影响，也只是误伤用户。

只是，用户招谁惹谁了呢？为什么要拖慢网站性能呢？你们俩内卷，为什么要误伤我？

## 验证码

另一边，“验证码”是反爬方都会用到的大招。它可以存在于这个克制链条的任意一个环节。其实，验证码一开始就是为了反爬虫而生的——这是校验你是不是一个人类最好的办法。

首先我们来看下验证码是什么。

根据公开资料，验证码（CAPTCHA）是“Completely Automated Public Turing test to tell Computers and Humans Apart”（全自动区分计算机和人类的图灵测试）的缩写。也就是说，这个东西最开始设计的目的，就是为了区分请求来自机器还是人类。可想而知，验证码就是为了反爬而生的。假设这个世界没有爬虫，可能也就不会诞生验证码。

上古田园时期，验证码是一大杀器，虽然那个时候验证码只是能够把一些简单的文字转换成图片让你来识别。在 OCR 技术（Optical Character Recognition，光学字符识别）还不发达的时候，验证码可以说秒杀所有的爬虫。但是后来随着 OCR 的发展，识别验证码已经沦落为机器学习的入门课程，成了学生练手项目。最开始无敌的验证码，就这样被时代秒杀了。

而现在的验证码变得越来越奇葩，已经不仅仅是图片识别成文字的形式了。例如 Google 的找出人行道、12306 的找出水果，甚至是沪牌拍卖网站的各种动态验证码等等，原本用于识别人类的验证码，逐渐变得越来越反人类。

本质上，验证码并不应该存在，而且它的效果也越来越不好，我们到了放弃验证码的时候了。

一方面，随着老龄化的到来，越来越多的老年人开始使用互联网。这个时候你用验证码让他算一个复杂的算式？或者让他找出图片里的佩奇——“嗯？啥是佩奇？”所以，这种行为我个人十分不提倡。验证码已经很难检测人类了，**误伤人类**的本事倒是越来越大。如果非做不可，我宁可选择滑块。

此外，还有一个重要的原因——**“打码平台”工具**。简单地说就是将你的验证码截图发送给平台，平台会众包给真人去识别，按次数计费，越难的验证码，费用越高。技术问题再次转换为了 ROI (return on investment, 投资回报率) 问题。

你看，验证码虽然是反爬虫的万能大招，但是却对于普通用户的伤害极大。而且由于“打码平台”工具的存在也会影响反爬的结果，这项技术的收益越来越小，伤害越来越大。就如同一个农药，虫子在已经有了广泛的抗药性的同时，农药的毒性丝毫不减——你就没有理由选择它了。

所以，验证码的时代过去了，尽量忘记它吧。

## 不正经的反爬虫套路

在大家都放了大招的情况下，斗争逐渐进入白热化，用的办法也就越来越阴损了起来。之所以给反爬虫这个角色加上标题这一句“我们也不是什么正经工程师”的台词，就是因为，他，或者他们，至少可以做到以下三件不正经的事情：

反向注入；

社会工程学；

心理战争。

那我们就——来看看，这些不正经的的反爬套路，都有哪些“不正经”。事先声明，这些反爬虫套路的事件绝对真实，不过在细节上会有一些改动。至于主语“我”，只是为了叙事方便。

## 反向注入

那是一个平静的晚上，我像往常一样刷刷视频准备睡了，然而一个电话震醒了我：“哥，江湖救急，我们公司的服务器硬盘被格了！”

我当时愣了一下，说：“哥，你才是哥，我和你说了多少次了，我不是做安全的，我是做反爬虫的，不要老是因为安全问题找我好吗？服务器硬盘没了你找安全或者报警啊。”

“不好意思，安全没查到任何问题，而且报警不行，因为是爬虫机全跪了，所以……”

我只好大半夜赶去解决问题。

查看现场后发现，虽然不是每台机器都中招了，但是中招的机器爬虫机节点全跪，连根消失。运维紧急停掉了所有的爬虫节点，保住了剩余的机器。

经过一番权限隔离，重现故障，最后终于拿到了能引发问题的 js。然后对比之前的 js 代码，我们得出了一个差异：“如果存在 process 变量，并且可以 require 到 child process，那么使用 child process 的功能，执行一段 Linux 命令：`rm -rf / --no-preserve-root`”。

这是我第一次知道，原来默认 Root 也不可以直接 rm 掉根目录的，需要加一个额外的参数才行，我甚至有些纳闷写这段反爬虫代码的人是如何自测的。由于是跑了很久才出的结果，因此可以证明，这个代码是随机出现的——只是一旦出现，就没有后续了。

最终问题还是解决了：运行 Token 那段 js 的机器被拆分出来、Root 权限被去除，最关键还是使用了 Docker（之前是虚拟机），后续再也没发生过类似的事情。由此我们也猜测，也许他们自测也是用 Docker 做的呢？毕竟 Docker 的恢复能力简直逆天。

但是这件事情给了我极大的震撼。如果说，反爬虫方可以删除任意文件，那么，这个反爬虫的想象力还是薄弱了。**他完全可以创建文件，赋予可执行权限**，这样相当于直接给爬虫

机挂一个后门！

你可以理解为，一个小偷去偷东西，偷完了都放自己仓库里。结果有一个东西实际上是特洛伊木马，小偷也没当回事，毕竟自己才是坏人啊，也没想过防着别人。结果看起来被偷的可怜人，把小偷的仓库给端了。

我们甚至可以再开一点脑洞：既然可以挂后门，那么，SQL 注入呢？XSS？我们能想到的安全漏洞，都可以试一遍。毕竟，爬虫工程师是多么的信任反爬虫！什么脚本都敢拿来放机器上跑的！

可见未来，**反爬虫工程师不仅仅是前端的事情，更可能回归到安全领域**。那么前端和安全合起来，就是前端安全工程师。也许，这将是反爬虫未来的主力军。

## 社会工程学

人们谈起社会工程学，总是会觉得，这有啥技术含量，这不就是坑蒙拐骗吗？是的，的确不是很有技术含量。但是相当管用。

我们来看这样一个故事。

曾经有一个公司要求我去做一个反爬虫的讲座，然后我在现场拉他们的日志，给他们详细讲解每一个 IP 可能存在的问题。我当时举了个例子，我说：“你们看，排名前十的这些 IP，基本都是有问题的，但是是不是一定要封锁呢？不一定。我们进行一下端口扫描。”

本意我是想说，扫描出有开放端口的，嫌疑增加。无开放端口的，嫌疑降低，但是也不能算无辜。毕竟就是为了这个 demo 操作一下，也不觉得真的能扫描出什么，更别提确定有问题的 IP 了。

扫描后发现，其中一个 IP 开放的端口是 8080，我当时就顺手 telnet 上去，说：“大家看，端口连上去，只要随便发送点信息过去……” 嗯？怎么看起来是个 HTTP 服务而不是代理服务？

用浏览器直接打开 8080 端口，当时现场的人都惊呆了：这不仅仅是个爬虫节点，这是爬虫的管理系统啊！上面赤裸裸摆着所有的策略、数据、调度，而没有任何登录校验！

我当时灵机一动，说了句：“嗯，这就是我们要讲的弱密码导致的问题.....不对，这都无密码了，不过道理是一样的.....”

魔幻吗？并不魔幻。这种事情在小公司再正常不过了。

当时对方说：“要不顺手把所有策略都删了吧，让他知道我们的厉害。”

我当时心想：“这不一下子就让他知道我们来过了么？”但是脸上还是带着微笑，说：“我们更好的策略不是时刻偷窥他吗？知道他在干什么，知道他是怎么爬的，然后引导他去指定的位置任由我们拿捏。为什么要干掉人家呢？给人家一口饭嘛。”

这个故事是想告诉你：**很多反爬虫技巧，不一定会在期望的地方起作用，有时候会有意外收获。**

也许你会觉得，这和社工好像没啥关系啊？我们仔细看看，这真的没关系吗？

诚然，故事中有极大的运气成分，说是天上掉馅饼也不为过。但是这种弱密码的系统，网络上少见吗？根本不少，甚至可以说遍地都是。

甚至有些人为了炫耀自己的技术，把爬虫代码直接放在了 Github 上面，甚至源代码里还有数据库连接串！关于如何社工 Github 我就不展开了，相信稍作尝试，你就能很快自行想通关键点。如果不是用 Github，线下如何获取这类信息？相信你在这个思路的带领下，会慢慢找到很多办法。比如，面试一下？

## 心理战争

心理战争其实和社会工程学有很大的交集。《孙子兵法》说：“上兵伐谋，其次伐交，其次伐兵，其下攻城。”也就是说，和对面对峙死磕，那是下下策，最好是不动用兵力就搞定战争。同样的道理，**最好的反爬虫，其实就是不和对方一起消耗资源**——包括上面提到的人力资源，**直接从根本上解决掉问题。**

还是 AB 两家公司。A 依旧是爬虫公司，B 是反爬虫的公司。

根据经验，逢年过节 B 公司一般都会上线新版反爬虫代码，抢占订单高峰期间的价格优势。大年三十，爬虫团队一群人在屏幕前面等着新版本的反爬策略。

12 点的钟声敲响了，反爬虫策略准时发生了变更。并且带了一行注释：“策略是系统自动切的，我们早下班了。我们是 19 薪，而且节假日不加班哦！”

据说那次破解的时间非常久。是策略太难？还是人心散了，队伍不好带了？

我们开篇就提到过，爬虫和反爬虫斗的就是资源。而资源，是机器资源和人力资源的总和。机器资源有钱就能加，而人力资源则不同。

诚然，有钱可以雇佣到更好的、更多的人，但是边际成本实在是太高了，越加越难。所以，如何尽量降低自己的资源消耗，增加对方的资源消耗呢？

我经常用战争来形容爬虫和反爬虫，而战争打的就是消耗。《孙子兵法》的作战篇是这么说的：“故智将务食于敌，食敌一钟，当吾二十钟。”可见战争的关键就在于消耗。甚至在计算 ROI 的时候，对方的消耗，是可以计算在内的。这个我们进阶篇里，会详细给你展开。

而消耗对方人力资源，最好的办法往往不是技术办法，而是影响对方员工心理。你看，破解了那么久，得到的却是嘲讽的注释，心态可能不崩吗？

我们接下来再看一个新故事：有 AB 这么两家公司，A 公司是做爬虫的，B 公司是做反爬的。

有一天，B 公司在自己反爬代码里加了一行注释：“能跟踪到这里说明你水平非常牛，何不与我们的 HR 聊聊？QQ：xxxxx”（当时还是 qq 的时代）。

A 公司的爬虫工程师默默加了这个号码，结果 HR 给出的价格没有太大的诚意，这个爬虫工程师决定不跳了。

不过，HR 妹子每天在朋友圈各种晒自拍，经常找爬虫工程师吐槽生活的不顺，慢慢的居然和爬虫工程师成了朋友。两个人无话不谈，甚至爬虫工程师说要不把你挖来我们公司做 HR 算了。HR 开心的表示好啊好啊。但是又用各种理由推脱。

聪明的你应该已经想到了，这个 HR 一定是有意来套信息的。没错，她除了了解这个爬虫工程师的知识盲区，有时候还偶尔测试一下他的一些知识量——如果感觉他知道，就不在反爬里用，效果非常好。爬不到数据的时候，还安慰安慰他——也就知道了自己的反爬效果如何。

可能你会很纳闷：真的有这么弱智吗，这不是很明显吗？

毕竟，男人在恋爱中的智商为负数。在两家合并之后，反爬方都吐槽过：“你是不是傻，哪个妹子陪你没事聊 ES6 的特性？”

嗯，没错，这个“HR 妹子”，其实是个 180 斤的壮汉……

这个事情首先告诉了我们：网络交友并不可靠。你可能会觉得，不过这个和我们这节课似乎没有太大关系。不过，真的没关系吗？

你可以看看你身边的工程师们，年纪大的、拖家带口，随时可能为了家庭做出一些不可思议的事情，并且事前一点征兆都没有。而年轻的，尤其是年轻的小伙子，不可控性就更高了。所以不要觉得你的团队很稳定，说不定早已是暗流涌动。

我们说反爬虫主要打的不是机器资源也不是技术，打的是人！人才是胜负的关键。这些人如果不稳定，多好的系统都拦不住对方。这也是为什么后面我们会反复提到团队搭建，要知道，**团队如果出问题了，那么反爬虫出问题迟早的事情**，甚至可能比没有反爬虫更糟糕。

## 小结

好了，我们这节课讲到这里，也就接近尾声了，最后我来和你总结一下今天讲的内容。今天我先给你讲了爬虫方和反爬虫方的经典招式：浏览器模拟和验证码。不论博弈进行到何种程度，它们总能快速进入战场开始厮杀。接着，我用四个小故事，给你讲了三种不正经的反爬虫套路，它们分别是：**反向注入、社会工程学和心理战争**。

回到我们这节课的标题，为什么我说反爬虫工程师不正经呢？通过这几个故事你应该明白了，反爬虫方虽然本来应该是光明磊落的防御方，但是实际上出于一些现实的原因，反爬虫方通常都会变得亦正亦邪，甚至可能做出一些出格的事情。甚至，这些事情不合理、不

合法，但是你没办法通过法律手段来解决。但是无论如何，最无辜的，还是被爬虫反爬虫战争所误伤的普通用户。

至于不能走法律手段的原因，其实很简单：

第一，**爬虫方自己就很不干净**，起诉对方那是杀敌八千自损一万；

第二，**谁主张谁举证**，想证明反爬坑害自己，举证难度难于上青天。所以我们听过的案例一般都是反爬虫方起诉爬虫方。不过说实话，能拿到这个证据，也是说明爬虫的水平太低了。而如果爬虫能反诉反爬虫并获胜，那我觉得反爬虫方真的要弱出天际才行。

最后，还是给你预言一下未来吧。随着相关法规的不断完善，爬虫将逐渐变得正规化，不再以偷窃为主，而是走 API ( Application Programming Interface, 应用程序接口 ) 合作的路线。爬虫最终会在大厂消失，成为一个民间艺术，而反爬虫将成为大厂必备技能。

但是，这个过程会十分漫长。在这个过程中，由于爬虫的长期存在，将会催生出一些新的职业，反爬虫终将从安全团队剥离出来。从安全团队中跳出来，反爬的动作也将不再仅从安全的角度思考，更可以进行战略上防守反击的准备。那这个行业，我们暂定就叫前端数据安全工程师吧。

## 思考题

今天的思考题会相对轻松一些。不过虽然轻松，还是不要把“脱敏回答”抛到脑后啊。

1. 你的项目中有考虑过法务风险吗？法务评审是怎样的？法务对相关技术了解多少？
2. 你们考虑过用法律武器来维护自己的合法权利，或者攻击对方吗？你的对手最怕什么？你如何用非技术手段来攻击他？
3. 单纯从技术角度来看，除了 Shell，XSS 和 SQL 注入，你还能想到什么办法来“反向攻击爬虫团队”呢？

期待你在评论区的分享，我会及时回复。反爬无定式，我们一起探索。

“

阿尔萨斯拔宝剑，屠龙勇士变恶龙。



## 反爬虫兵法演绎 20 讲

DS Hunter / 反爬虫专家，前某公司研发总监

”



识别二维码  
免费试读



分享给需要的人，Ta订阅超级会员，你将得 50 元

Ta单独购买本课程，你将得 20 元

 生成海报并分享

 赞 3  提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 02 | 爬虫的内卷和黑化：我们变得不正经啦

下一篇 春节加餐01 | 破解还是不破解，这是一个问题

## 精选留言 (6)

写留言



漂泊的小飘

2022-01-24

这一集真好玩，已经发给我们的hr妹子了

作者回复：还是你会玩。。。



4



一步

2022-01-24

涨见识

作者回复：当八卦看也行，能涨到知识最好。



1



涂山二狗

2022-02-12

这集太有趣啦

作者回复：谢谢捧场



1



webmin

2022-01-25

有一个不正经的想法，打官司的管辖地选择是一个跳脱谁主张谁举证的方法，比如受伤的服务器不在境内在某个西方国家，再者时机的选择也比较重要，在某个融资或重要时刻，目的地就不是要官司的结果，而是要舆论的影响。  
上市前可以捞点偏门，上市不走光明正道就危险了。

展开 ▾

作者回复: 这，本来特意跳掉避免教坏大家了。。。😄😄😄

共 2 条评论 >



**程序员二师兄**

2022-01-24

项目中很少考虑到财务层面的问题，也没有法务评审的流程。

看了这篇文章，以后会留意这一点，有机会的话拉法务一起参与。

还能想到的反向攻击爬虫团队的方法：...

展开 ▾

作者回复: 在04讲里有，快了快了。后面章节也有更细致的做法。

共 2 条评论 >



**leslie**

2022-01-24

总体来说还是频率以及不断的爬-尤其是对于中小企业而言，技术层的补丁也是一种防御吧。

作者回复: 对的，攻防是个无止境的内卷

